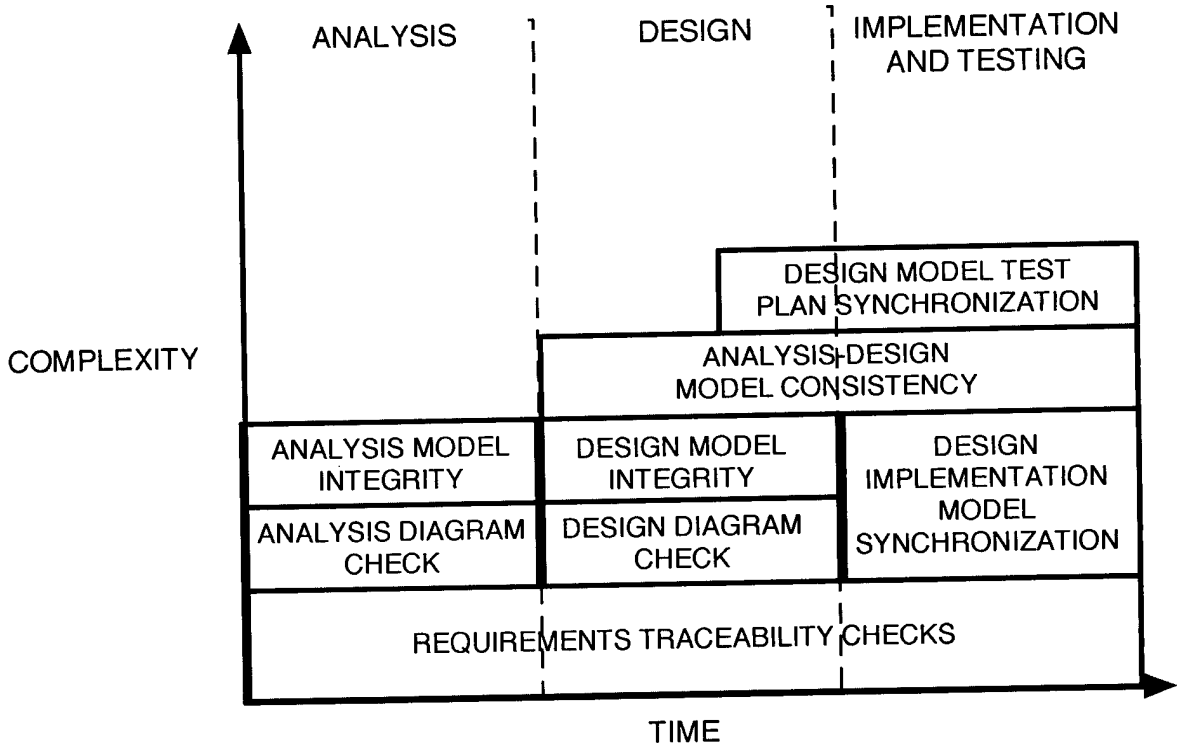
ANALYSIS
DESIGN
IMPLEMENTATION AND TESTING
COMPLEXITY
DESIGN MODEL TEST PLAN SYNCHRONIZATION
ANALYSIS-DESIGN MODEL CONSISTENCY
ANALYSIS MODEL INTEGRITY
DESIGN MODEL INTEGRITY
DESIGN IMPLEMENTATION MODEL SYNCHRONIZATION
ANALYSIS DIAGRAM CHECK
DESIGN DIAGRAM CHECK
REQUIREMENTS TRACEABILITY CHECKS
TIME

FIGURE 1

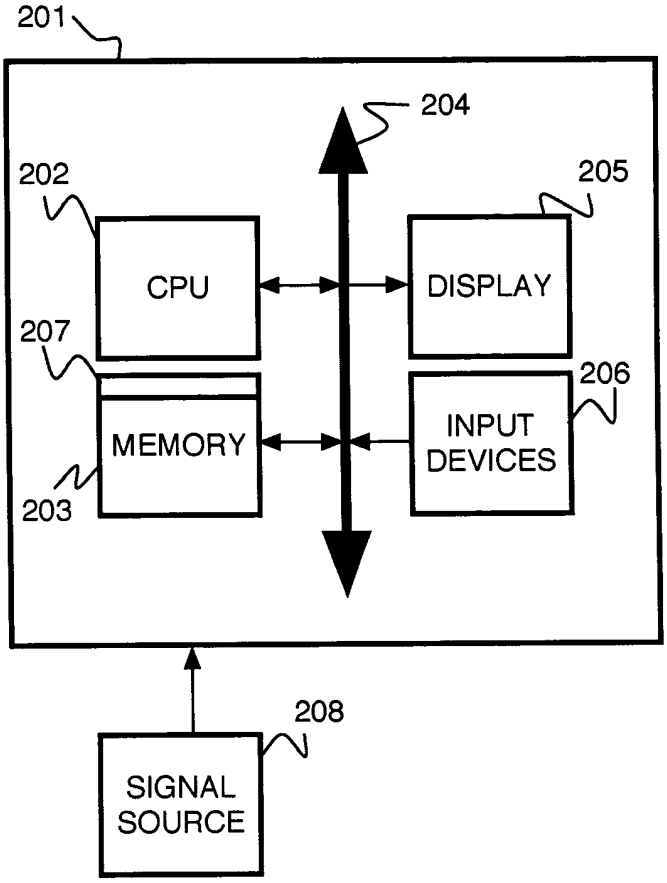
201
204
202
205
CPU
DISPLAY
207
206
MEMORY
INPUT
DEVICES
203
208
SIGNAL
SOURCE

FIGURE 2

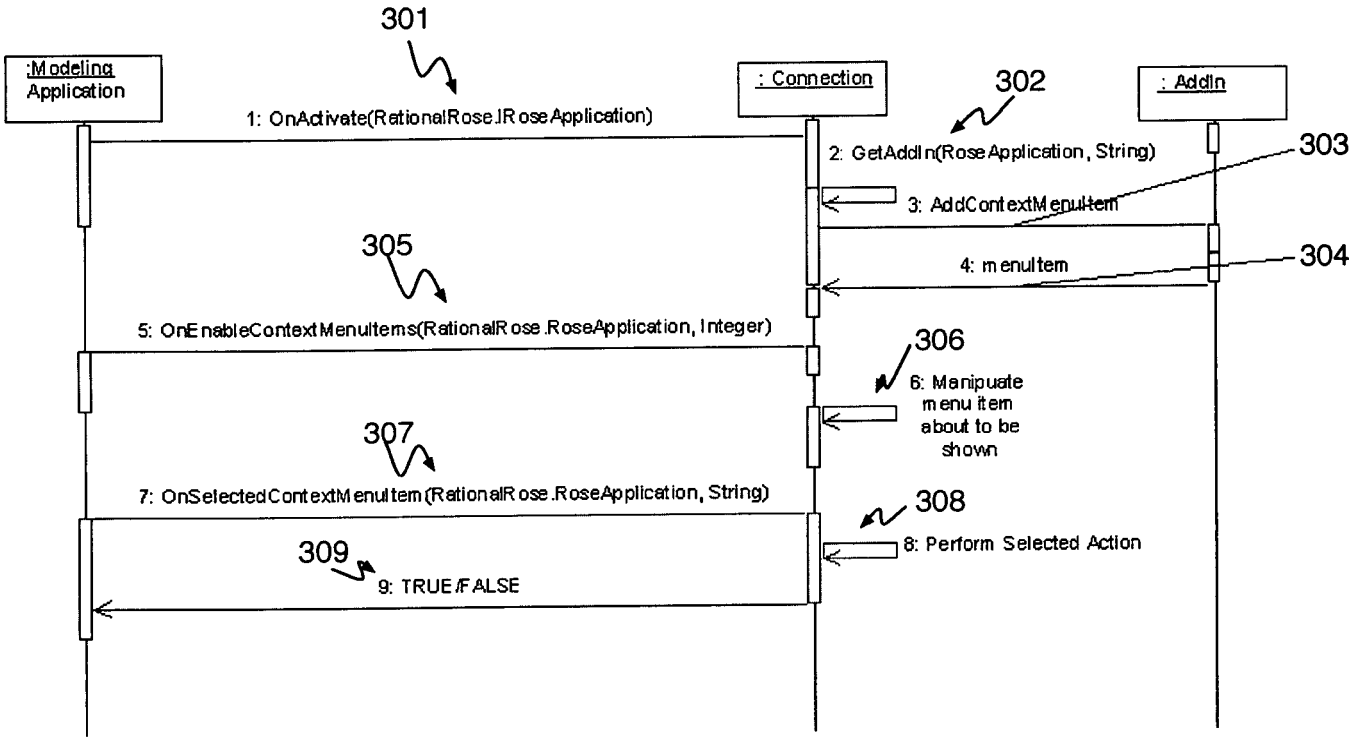
301
:Modeling Application
: Connection
302
: AddIn
1: OnActivate(RationalRose.IRoseApplication)
2: GetAddIn(RoseApplication, String)
303
3: AddContextMenuItem
305
4: menuItem
304
5: OnEnableContextMenuItems(RationalRose.RoseApplication, Integer)
306
6: Manipuate menu item about to be shown
307
7: OnSelectedContextMenuItem(RationalRose.RoseApplication, String)
308
8: Perform Selected Action
309
9: TRUE/FALSE

FIGURE 3

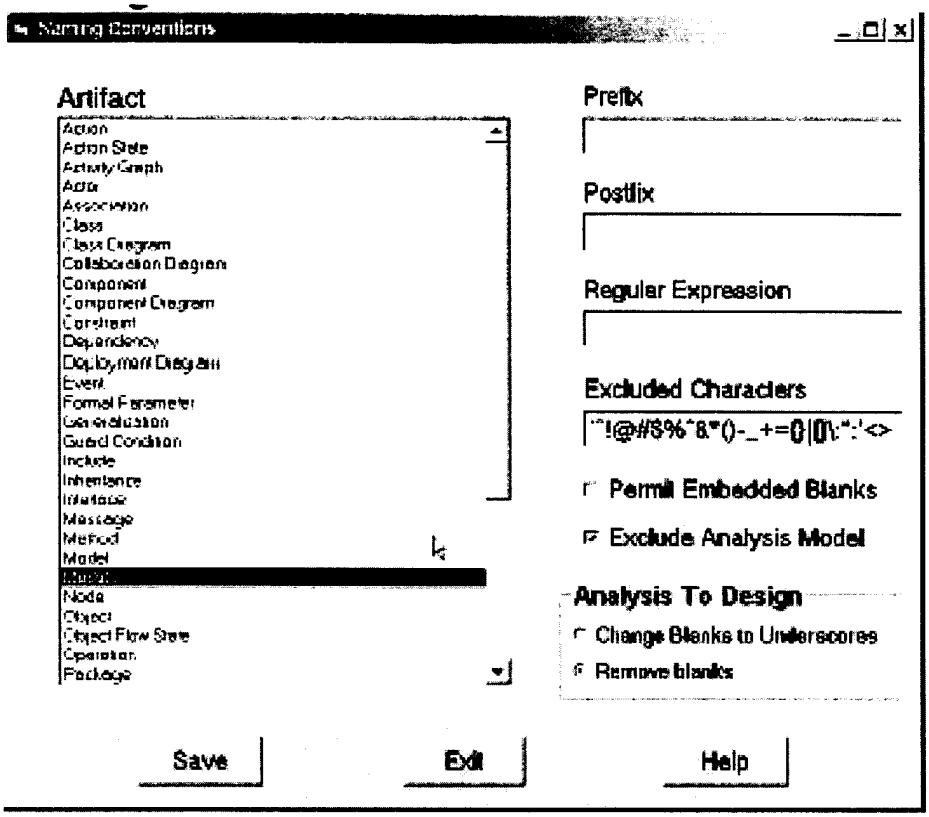
Naming Conventions
Artifact
Action
Action State
Activity Graph
Actor
Association
Class
Class Diagram
Collaboration Diagram
Component
Component Diagram
Constraint
Dependency
Deployment Diagram
Event
Formal Parameter
Generalization
Guard Condition
Include
Inheritance
Interface
Message
Method
Model
Node
Object
Object Flow State
Operation
Package
Prefix
Postfix
Regular Expression
Excluded Characters
`~!@#$%^&*()-_+={}|[]\:";'<>
Permit Embedded Blanks
Exclude Analysis Model
Analysis To Design
Change Blanks to Underscores
Remove blanks
Save
Exit
Help

FIGURE 4A

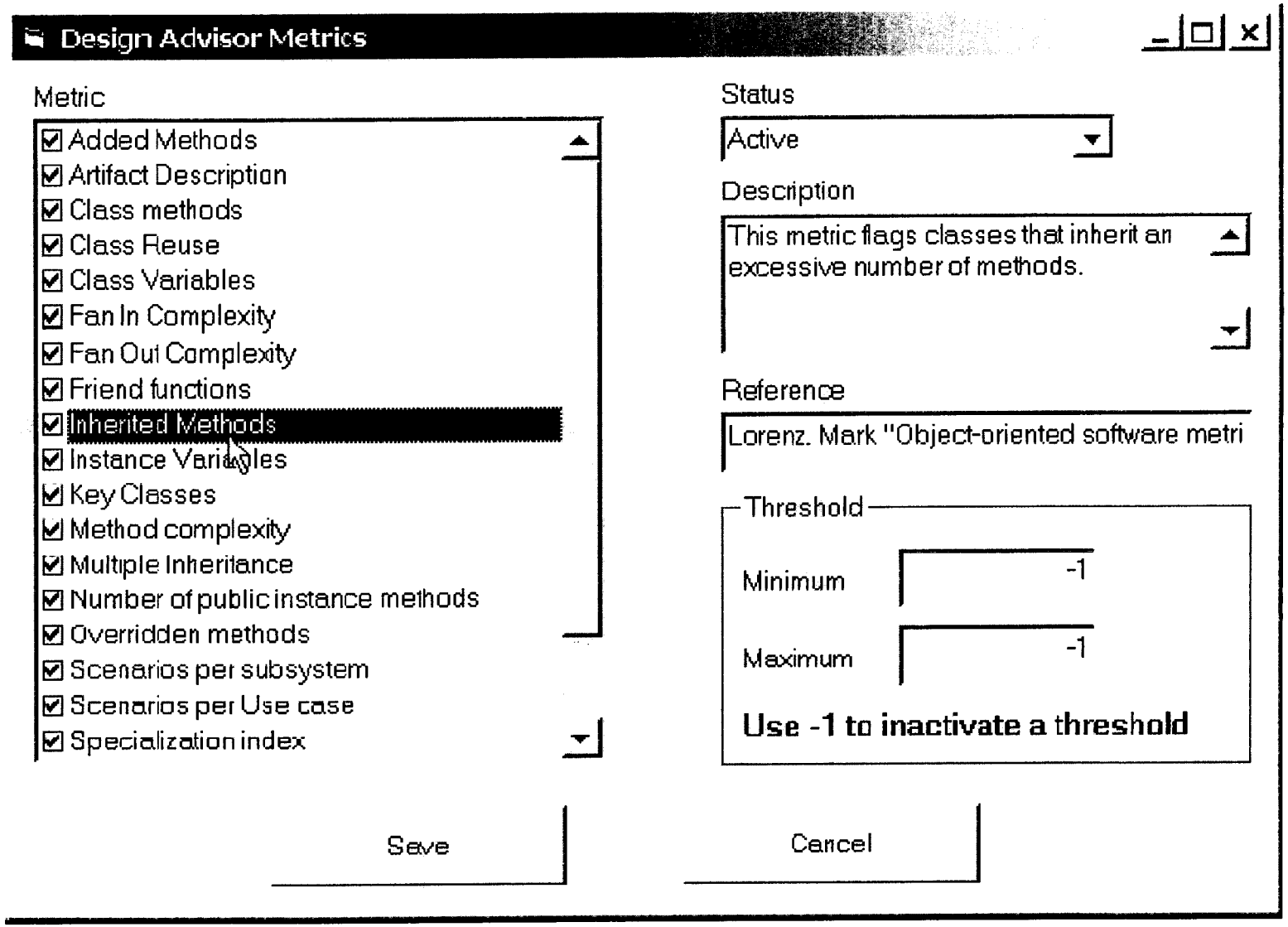
Design Advisor Metrics
Metric
Added Methods
Artifact Description
Class methods
Class Reuse
Class Variables
Fan In Complexity
Fan Out Complexity
Friend functions
Inherited Methods
Instance Variables
Key Classes
Method complexity
Multiple Inheritance
Number of public instance methods
Overridden methods
Scenarios per subsystem
Scenarios per Use case
Specialization index
Status
Active
Description
This metric flags classes that inherit an excessive number of methods.
Reference
Lorenz. Mark "Object-oriented software metri
Threshold
Minimum
-1
Maximum
-1
Use -1 to inactivate a threshold
Save
Cancel

FIGURE 4B

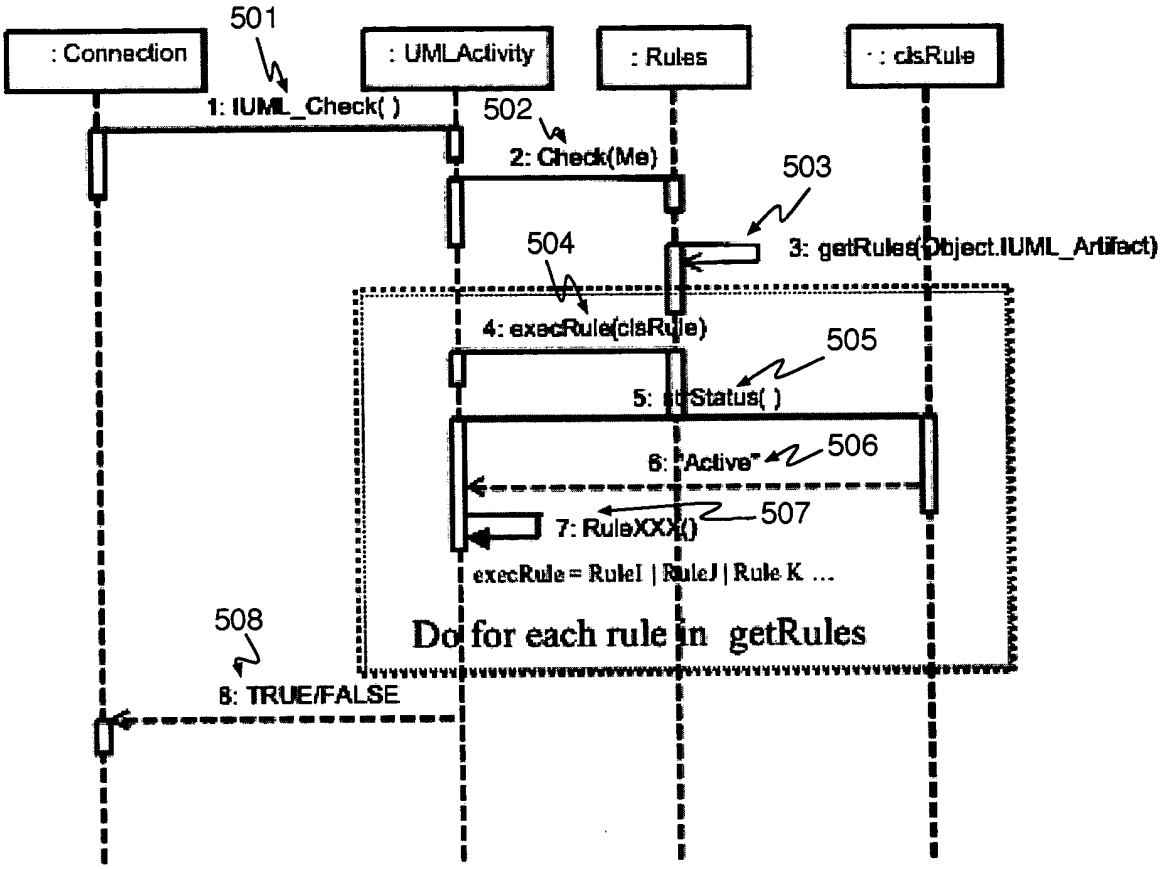
501
: Connection
: UMLActivity
: Rules
: clsRule
1: IUML_Check( )
502
2: Check(Me)
503
3: getRules(Object.IUML_Artifact)
504
4: execRule(clsRule)
505
5: Status( )
6: "Active"
506
7: RuleXXX()
507
execRule = RuleI | RuleJ | Rule K ...
Do for each rule in getRules
508
8: TRUE/FALSE

FIGURE 5A

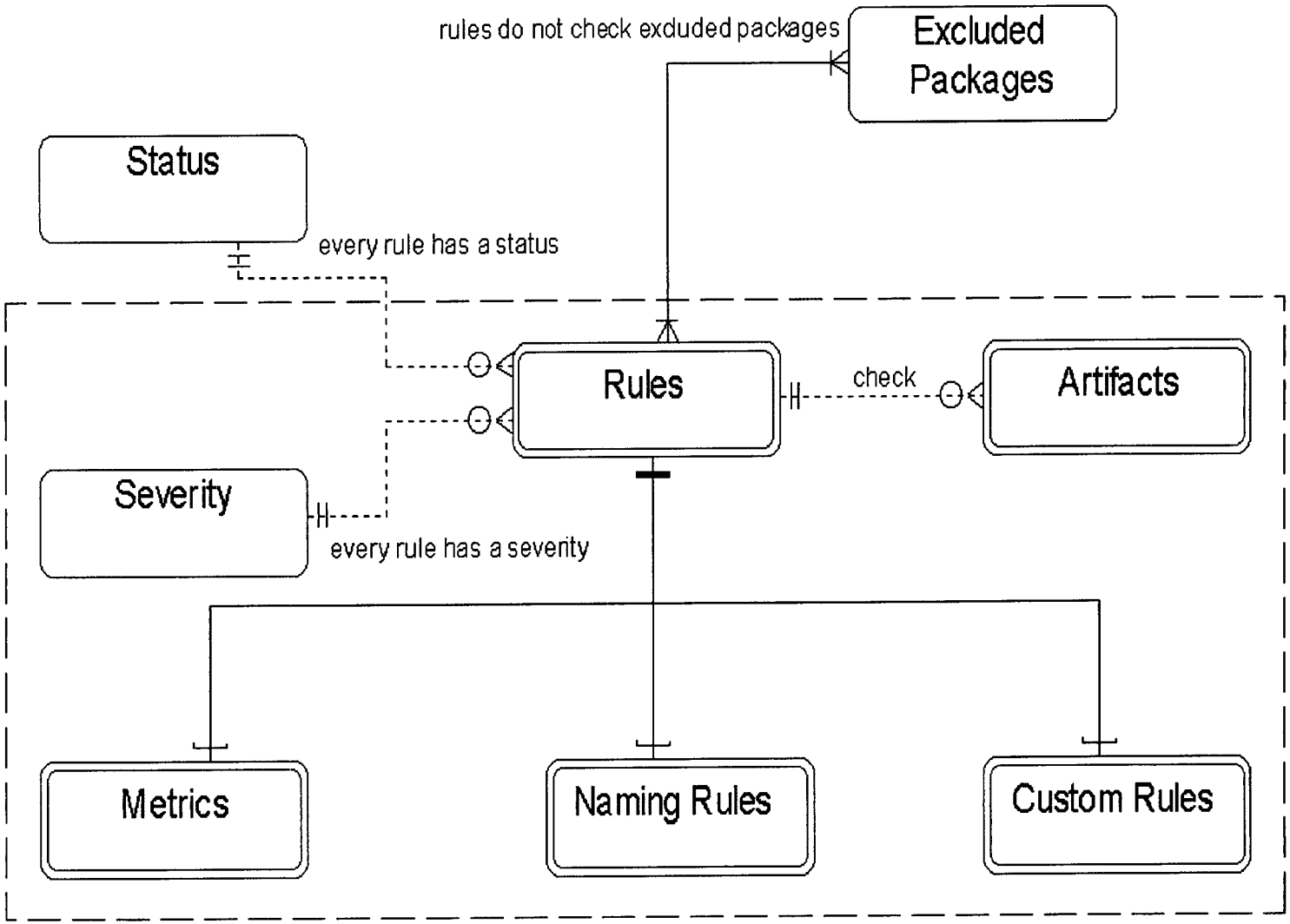
rules do not check exduded packages
Excluded Packages
Status
every rule has a status
Rules
check
Artifacts
Severity
every rule has a severity
Metrics
Naming Rules
Custom Rules

FIGURE 5B

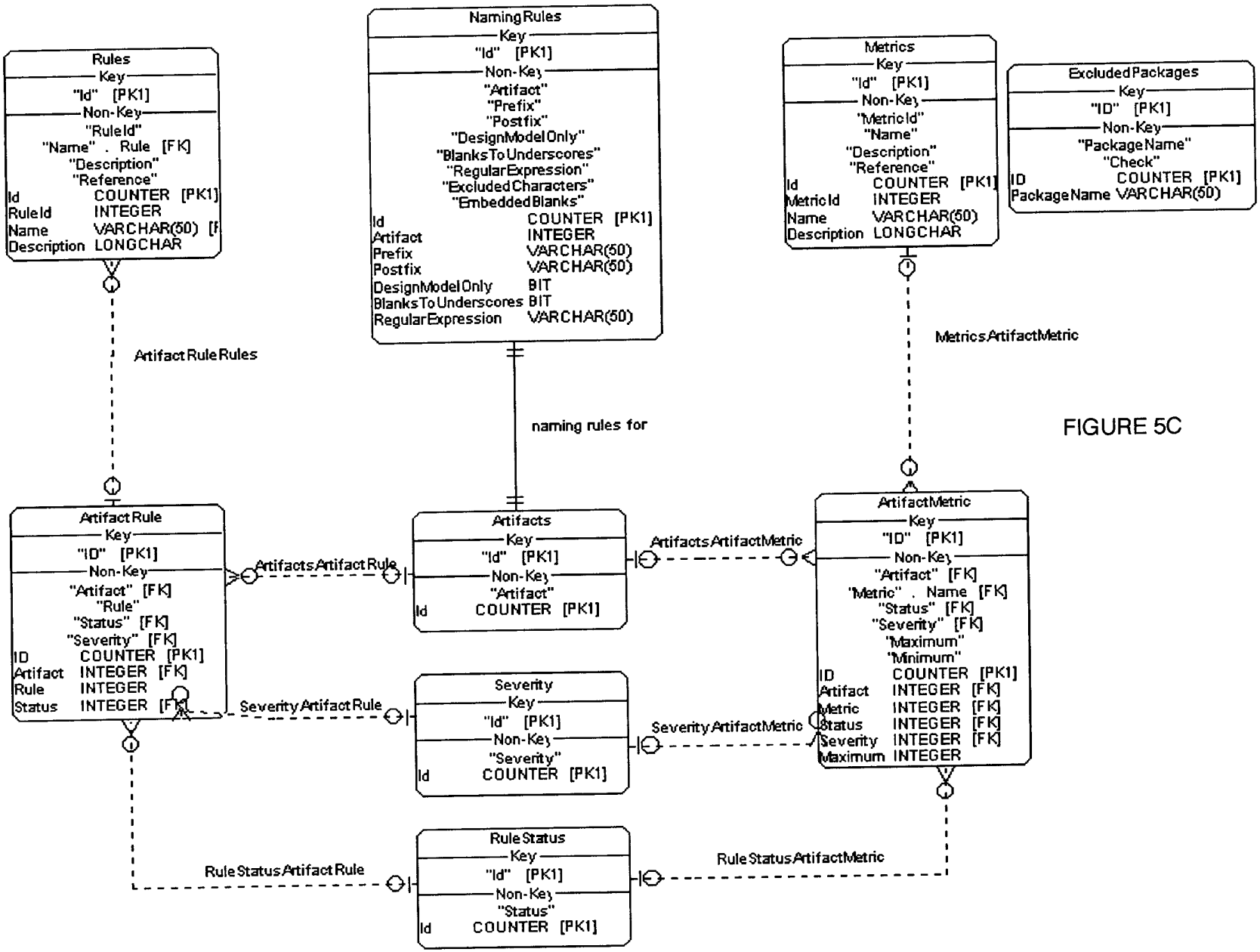
Rules
Key
"Id" [PK1]
Non-Key
"RuleId"
"Name" . Rule [FK]
"Description"
"Reference"
Id COUNTER [PK1]
RuleId INTEGER
Name VARCHAR(50) [F
Description LONGCHAR
NamingRules
Key
"Id" [PK1]
Non-Key
"Artifact"
"Prefix"
"Postfix"
"DesignModelOnly"
"BlanksToUnderscores"
"RegularExpression"
"ExcludedCharacters"
"EmbeddedBlanks"
Id COUNTER [PK1]
Artifact INTEGER
Prefix VARCHAR(50)
Postfix VARCHAR(50)
DesignModelOnly BIT
BlanksToUnderscores BIT
RegularExpression VARCHAR(50)
Metrics
Key
"Id" [PK1]
Non-Key
"MetricId"
"Name"
"Description"
"Reference"
Id COUNTER [PK1]
MetricId INTEGER
Name VARCHAR(50)
Description LONGCHAR
ExcludedPackages
Key
"ID" [PK1]
Non-Key
"PackageName"
"Check"
ID COUNTER [PK1]
PackageName VARCHAR(50)
ArtifactRuleRules
naming rules for
MetricsArtifactMetric
ArtifactRule
Key
"ID" [PK1]
Non-Key
"Artifact" [FK]
"Rule"
"Status" [FK]
"Severity" [FK]
ID COUNTER [PK1]
Artifact INTEGER [FK]
Rule INTEGER
Status INTEGER [FK]
ArtifactsArtifactRule
Artifacts
Key
"Id" [PK1]
Non-Key
"Artifact"
Id COUNTER [PK1]
ArtifactsArtifactMetric
ArtifactMetric
Key
"ID" [PK1]
Non-Key
"Artifact" [FK]
"Metric" . Name [FK]
"Status" [FK]
"Severity" [FK]
"Maximum"
"Minimum"
ID COUNTER [PK1]
Artifact INTEGER [FK]
Metric INTEGER [FK]
Status INTEGER [FK]
Severity INTEGER [FK]
Maximum INTEGER
SeverityArtifactRule
Severity
Key
"Id" [PK1]
Non-Key
"Severity"
Id COUNTER [PK1]
SeverityArtifactMetric
RuleStatusArtifactRule
RuleStatus
Key
"Id" [PK1]
Non-Key
"Status"
Id COUNTER [PK1]
RuleStatusArtifactMetric

FIGURE 5C

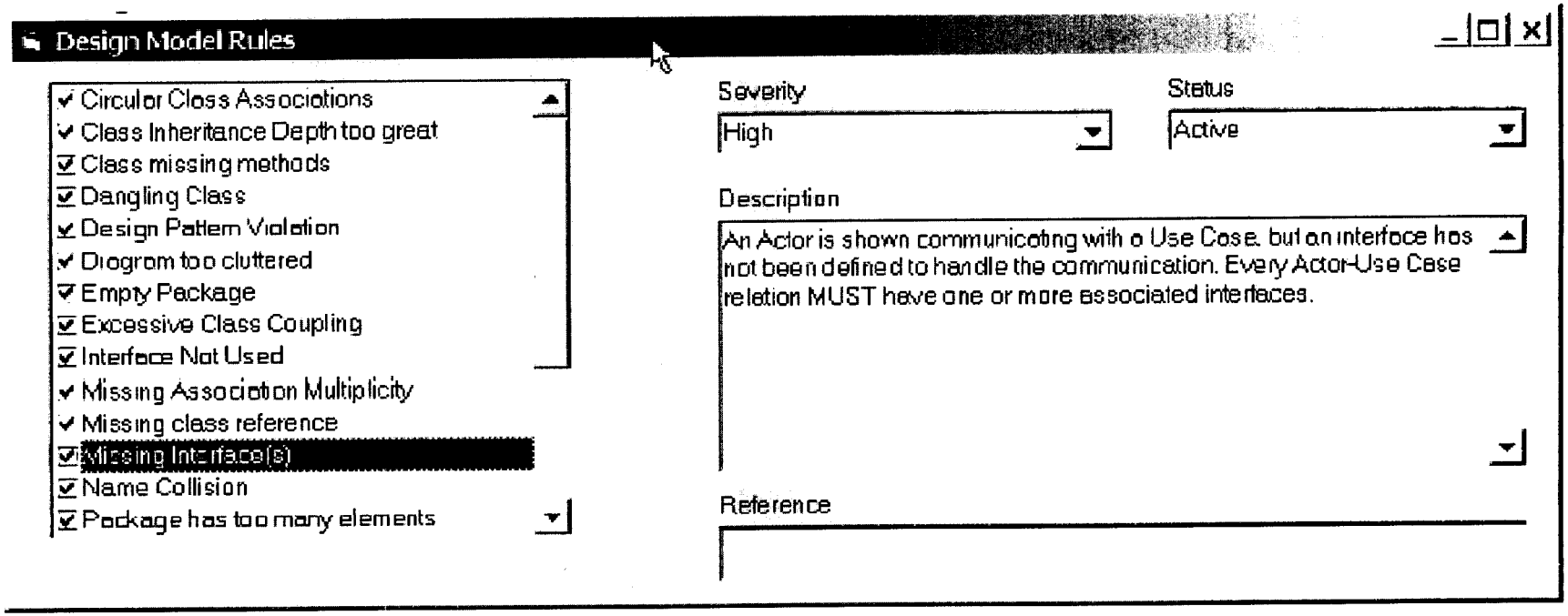
Design Model Rules
Circular Class Associations
Class Inheritance Depth too great
Class missing methods
Dangling Class
Design Pattern Violation
Diogram too cluttered
Empty Package
Excessive Class Coupling
Interface Not Used
Missing Association Multiplicity
Missing class reference
Missing Interface(s)
Name Collision
Package has too many elements
Severity
High
Status
Active
Description
An Actor is shown communicating with a Use Case, but an interface has not been defined to handle the communication. Every Actor-Use Case relation MUST have one or more associated interfaces.
Reference

FIGURE 6

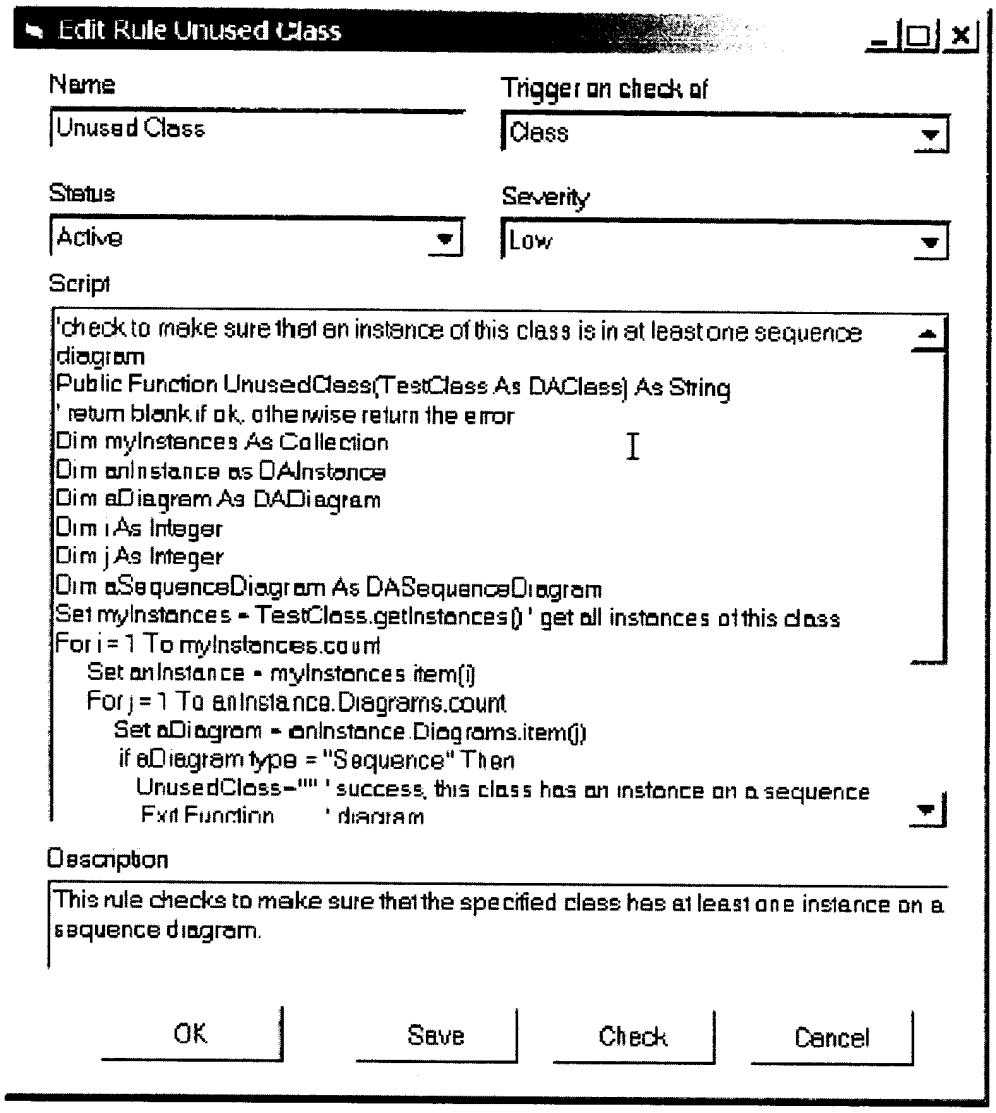
Edit Rule Unused Class
Name
Unused Class
Trigger on check of
Class
Status
Active
Severity
Low
Script
'check to make sure that an instance of this class is in at least one sequence diagram
Public Function UnusedClass(TestClass As DAClass) As String
' return blank if ok, otherwise return the error
Dim myInstances As Collection
Dim anInstance as DAInstance
Dim aDiagram As DADiagram
Dim i As Integer
Dim j As Integer
Dim aSequenceDiagram As DASequenceDiagram
Set myInstances = TestClass.getInstances() ' get all instances of this class
For i = 1 To myInstances.count
Set anInstance = myInstances.item(i)
For j = 1 To anInstance.Diagrams.count
Set aDiagram = anInstance.Diagrams.item(j)
if aDiagram.type = "Sequence" Then
UnusedClass="" ' success, this class has an instance on a sequence
Exit Function ' diagram
Description
This rule checks to make sure that the specified class has at least one instance on a sequence diagram.
OK
Save
Check
Cancel

FIGURE 7

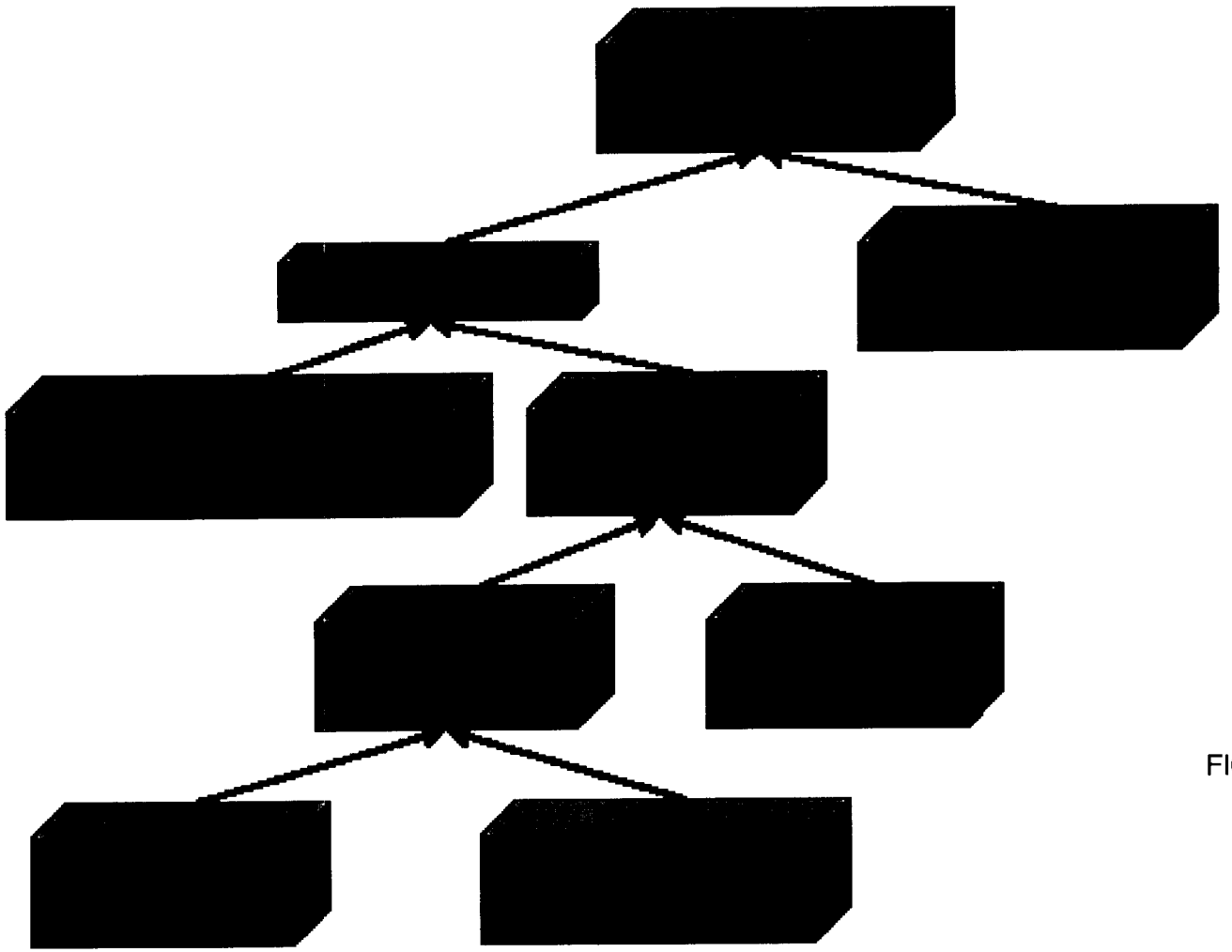

FIGURE 8

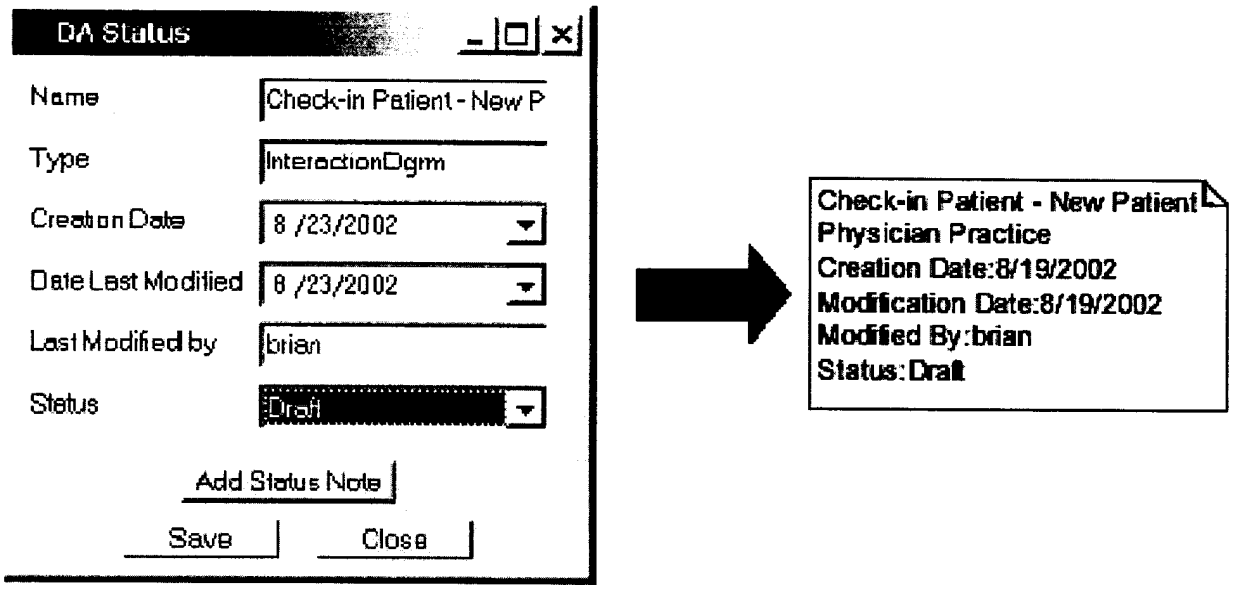
DA Status
Name
Check-in Patient - New P
Type
InteractionDgm
Creation Date
8 /23/2002
Date Last Modified
8 /23/2002
Last Modified by
brian
Status
Draft
Add Status Note
Save
Close
Check-in Patient - New Patient
Physician Practice
Creation Date:8/19/2002
Modification Date:8/19/2002
Modified By:brian
Status:Draft

FIGURE 9

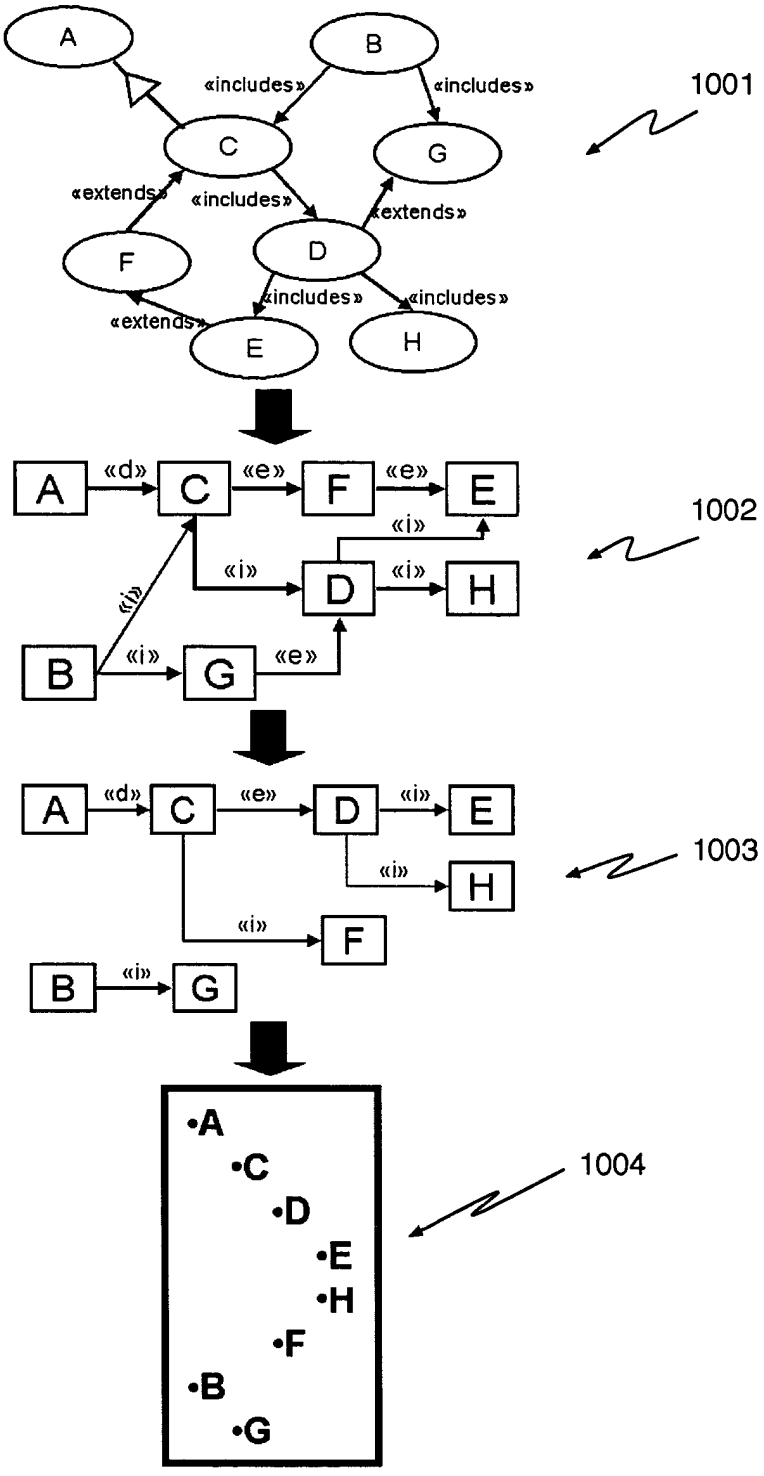
A
B
«includes»
«includes»
1001
C
G
«extends»
«includes»
«extends»
F
D
«includes»
«includes»
«extends»
E
H
A
«d»
C
«e»
F
«e»
E
«i»
1002
«i»
«i»
D
«i»
H
B
«i»
G
«e»
A
«d»
C
«e»
D
«i»
E
«i»
1003
H
«i»
F
B
«i»
G
•A
•C
•D
•E
•H
•F
•B
•G
1004

FIGURE 10

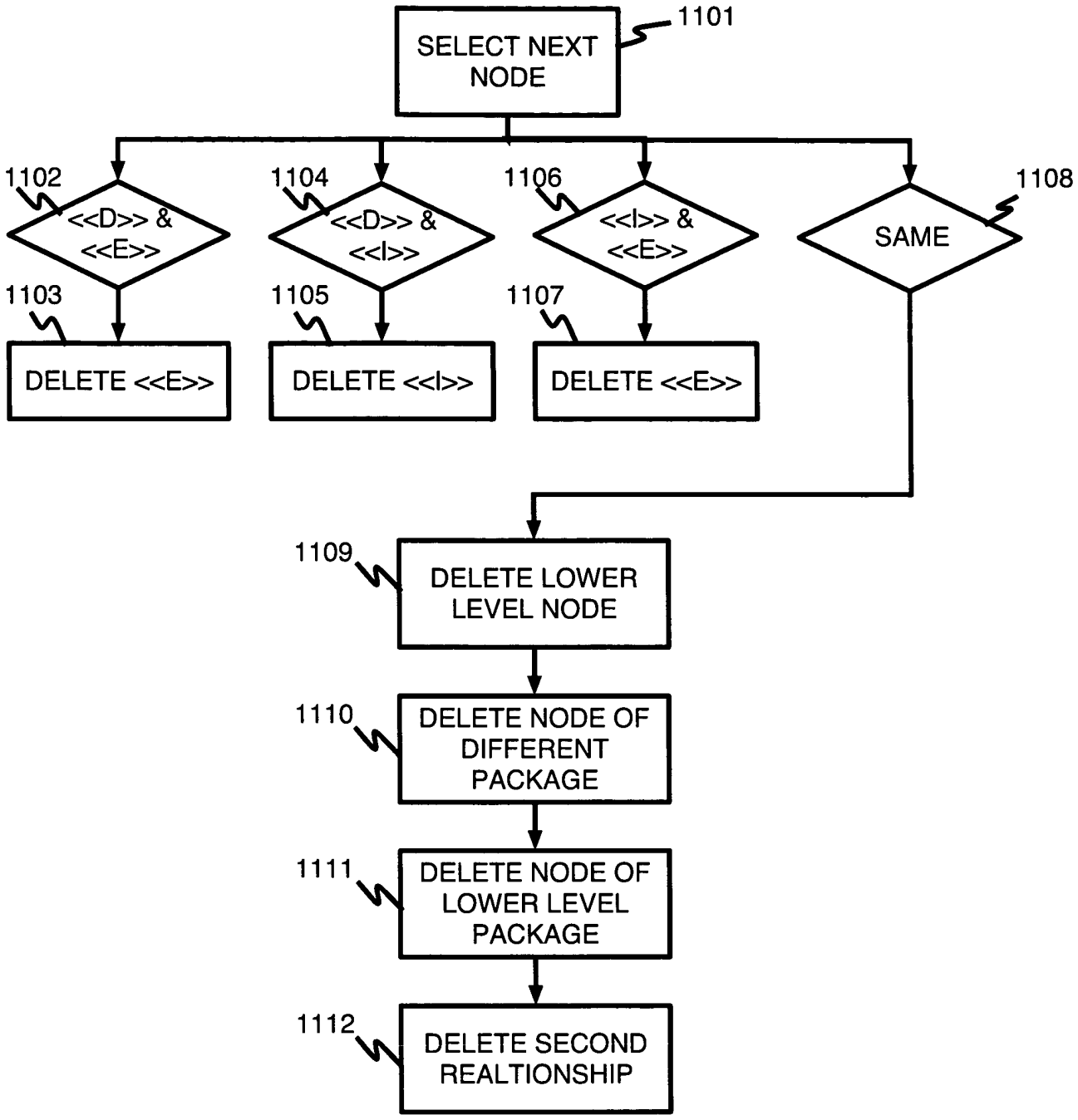
SELECT NEXT NODE
1101
1102
<<D>> & <<E>>
1103
DELETE <<E>>
1104
<<D>> & <<I>>
1105
DELETE <<I>>
1106
<<I>> & <<E>>
1107
DELETE <<E>>
1108
SAME
1109
DELETE LOWER LEVEL NODE
1110
DELETE NODE OF DIFFERENT PACKAGE
1111
DELETE NODE OF LOWER LEVEL PACKAGE
1112
DELETE SECOND REALTIONSHIP

FIGURE 11

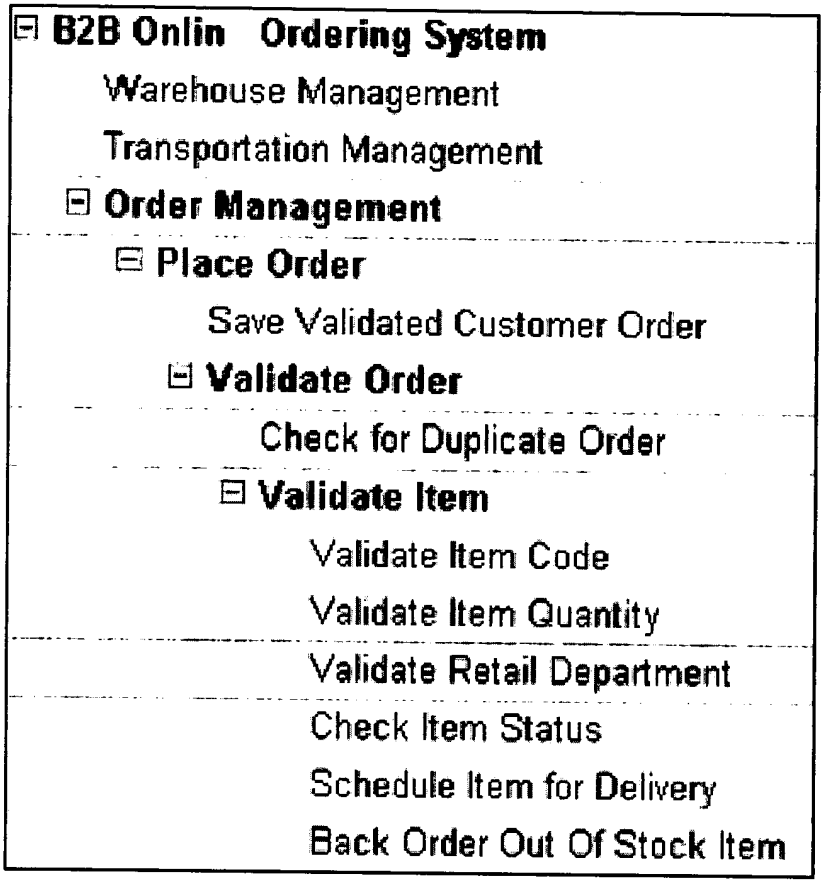
B2B Onlin Ordering System
Warehouse Management
Transportation Management
Order Management
Place Order
Save Validated Customer Order
Validate Order
Check for Duplicate Order
Validate Item
Validate Item Code
Validate Item Quantity
Validate Retail Department
Check Item Status
Schedule Item for Delivery
Back Order Out Of Stock Item

FIGURE 12